Package 'streamR'

January 5, 2014

Title Access to Twitter Streaming API via R

Description This package provides a series of functions that allow R users
to access Twitter's filter, sample, and user streams, and to parse the output into data frames.

Version 0.2

Author Pablo Barbera <pablo.barbera@nyu.edu>

Maintainer Pablo Barbera <pablo.barbera@nyu.edu>

Depends R (>= 2.12.0), RCurl, rjson, ROAuth

Suggests rmongodb

License GPL-2

Collate 'filterStream.R' 'parseTweets.R' 'sampleStream.R'
'userStream.R' 'streamR-package.R' 'countTweets.R' 'topRetweets.R' 'getTweets.R' 'readTweets.R'

R topics documented:

streamR-package 2
countTweets 2
example_tweets 3
filterStream 4
getTweets 6
parseTweets 8
readTweets 9
sampleStream 10
topRetweets 12
userStream 13

Index **16**

streamR-package *Access to Twitter Streaming APIs via R*

Description

This package provides a series of functions that allow R users to access Twitter's filter, sample, and user streams, and to parse the output into data frames or mongoDB.

Author(s)

Pablo Barbera <pablo.barbera@nyu.edu>

See Also

filterStream, sampleStream, userStream, readTweets, parseTweets getTweets, countTweets, topRetweets

countTweets *Connect to Mongo database and return count of tweets that match certain conditions.*

Description

countTweets opens a connection to a Mongo database and returns the number of tweets that match a series of conditions: whether it contains a certain keyword, whether it is or not a retweet, or whether or not it contains a hashtag. It depends on the rmongodb package, and a mongo object needs to be loaded in the workspace.

Usage

```
countTweets(ns, string = NULL, retweets = NULL,
  hashtags = NULL, verbose = TRUE)
```

Arguments

ns	string, namespace of the MongoDB collection where tweets were stored. Generally, it will be of the form "database.collection".
string	string, set to NULL by default (will return count of all tweets). If it is a string, it will return the number of tweets that contain that string in the text of the tweet.
retweets	logical, set to NULL by default (will return count of all tweets). If TRUE, will count only tweets that are retweets (i.e. contain an embededed retweeted status - manual retweets are not included). If FALSE, will count only tweets that are not retweets (manual retweets are now included).
hashtags	logical, set to NULL by default (will return count of all tweets). If TRUE, will count only tweets that use a hashtag. If FALSE, will count only tweets that do not contain a hashtag.
verbose	logical, default is TRUE, which generates some output to the R console with information about the count of tweets. If codeFALSE, function will not return any object.

Author(s)

Pablo Barbera <pablo.barbera@nyu.edu>

Examples

```
## Not run:
## capture 100 tweets that mention "twitter" and store them in local MongoDB,
## in collection twitter of database tweets
## (Note: since the track parameter searches also in URLs, this search
## is equivalent to capturing only tweets that contain pictures sent
## using Twitters built-in service)
 load(my_oauth)
 tweets <- filterStream( ns="tweets.twitter",
      track="twitter", tweets=100, oauth=my_oauth )

## connect to the Mongo database using rmongodb package
 library(rmongodb)
 mongo <- mongo.create("localhost", db="tweets")
## if required, specify username and password
## (MongoDB defaults are empty username and password)
 mongo.authenticate(mongo, username="", password="",
   db="tweets")

## count all tweets in the database
 countTweets( ns="tweets.twitter" )

## count tweets that mention twitter in their text
 countTweets( ns="tweets.twitter", string="twitter")

## count all retweets in the database
 countTweets( ns="tweets.twitter", retweets=TRUE)

## count all tweets that mention twitter and are not retweets
 countTweets( ns="tweets.twitter", string="twitter", retweets=FALSE)

## count all tweets that use a hashtag
 countTweets( ns="tweets.twitter", hashtags=TRUE)

## End(Not run)
```

`example_tweets` *Ten sample tweets published by @twitterapi*

Description

A vector of string characters that contains ten sample tweets in plain text.

Source

http://www.twitter.com/twitterapi

`filterStream` *Connect to Twitter Streaming API and return public statuses that match one or more filter predicates.*

Description

`filterStream` opens a connection to Twitter's Streaming API that will return public statuses that match one or more filter predicates. Tweets can be filtered by keywords, users, and location. The output can be saved as an object in memory, written to a text file or stored in MongoDB

Usage

```
filterStream(file.name = NULL, track = NULL,
  follow = NULL, locations = NULL, language = NULL,
  timeout = 0, tweets = NULL, oauth, ns = NULL,
  host = "localhost", username = "", password = "",
  verbose = TRUE)
```

Arguments

`file.name`	string, name of the file where tweets will be written. "" indicates output to the console, which can be redirected to an R object (see examples). If the file already exists, tweets will be appended (not overwritten).
`track`	string or string vector containing keywords to track. See the `track` parameter information in the Streaming API documentation for details: `http://dev.twitter.com/docs/streaming-apis/parameters#track`.
`follow`	string or numeric, vector of Twitter user IDs, indicating the users whose public statuses should be delivered on the stream. See the `follow` parameter information in the Streaming API documentation for details: `http://dev.twitter.com/docs/streaming-apis/parameters#follow`.
`locations`	numeric, a vector of longitude, latitude pairs (with the southwest corner coming first) specifying sets of bounding boxes to filter public statuses by. See the `locations` parameter information in the Streaming API documentation for details: `http://dev.twitter.com/docs/streaming-apis/parameters#locations`
`language`	string or string vector containing a list of BCP 47 language identifiers. If not `NULL` (default), function will only return tweets that have been detected as being written in the specified languages. Note that this parameter can only be used in combination with any of the other filter parameters. See documentation for details: `https://dev.twitter.com/docs/streaming-apis/parameters#language`
`timeout`	numeric, maximum length of time (in seconds) of connection to stream. The connection will be automatically closed after this period. For example, setting `timeout` to 10800 will keep the connection open for 3 hours. The default is 0, which will keep the connection open permanently.
`tweets`	numeric, maximum number of tweets to be collected when function is called. After that number of tweets have been captured, function will stop. If set to `NULL` (default), the connection will be open for the number of seconds specified in `timeout` parameter.
`oauth`	an object of class `oauth` that contains the access tokens to the user's twitter session. This is currently the only method for authentication. See examples for more details.

`ns`	string, namespace of the collection to which tweets will be added. Generally, it will be of the form "database.collection". If the database or the collection do not exist, they will be automatically created; if they exist, tweets will be appended.
`host`	string host/port where mongo database is hosted. Default is localhost (127.0.0.1).
`username`	string, username to be used for authentication purposes with MongoDB.
`password`	string, password corresponding to the given username.
`verbose`	logical, default is `TRUE`, which generates some output to the R console with information about the capturing process.

Details

`filterStream` provides access to the statuses/filter Twitter stream.

It will return public statuses that match the keywords given in the `track` argument, published by the users specified in the `follow` argument, and sent within the location bounding boxes declared in the `locations` argument.

Note that location bounding boxes do not act as filters for other filter parameters. In the fourth example below, we capture all tweets containing the term rstats (even non-geolocated tweets) OR coming from the New York City area. For more information on how the Streaming API request parameters work, check the documentation at: `http://dev.twitter.com/docs/streaming-apis/parameters`.

If any of these arguments is left empty (e.g. no user filter is specified), the function will return all public statuses that match the other filters. At least one predicate parameter must be specified.

Note that when no file name is provided, tweets are written to a temporary file, which is loaded in memory as a string vector when the connection to the stream is closed.

The total number of actual tweets that are captured might be lower than the number of tweets requested because blank lines, deletion notices, and incomplete tweets are included in the count of tweets downloaded.

To store tweets in MongoDB, it is necessary to install the MongoDB server in a local or remote machine. See here for instructions: `http://docs.mongodb.org/manual/installation/`

Author(s)

Pablo Barbera <`pablo.barbera@nyu.edu`>

See Also

`sampleStream`, `userStream`, `parseTweets`

Examples

```
## Not run:

## An example of an authenticated request using the ROAuth package,
## where consumerkey and consumer secret are fictitious.
## You can obtain your own at dev.twitter.com
  library(ROAuth)
  requestURL <- "https://api.twitter.com/oauth/request_token"
  accessURL <- "http://api.twitter.com/oauth/access_token"
  authURL <- "http://api.twitter.com/oauth/authorize"
  consumerKey <- "xxxxxyyyyyzzzzzz"
  consumerSecret <- "xxxxxxyyyyyzzzzzzz111111222222"
```

```
    my_oauth <- OAuthFactory$new(consumerKey=consumerKey,
      consumerSecret=consumerSecret, requestURL=requestURL,
      accessURL=accessURL, authURL=authURL)
    my_oauth$handshake(cainfo = system.file("CurlSSL", "cacert.pem", package = "RCurl"))
    filterStream( file="tweets_rstats.json",
     track="rstats", timeout=3600, oauth=my_oauth )

  ## capture 10 tweets mentioning the "Rstats" hashtag
    filterStream( file.name="tweets_rstats.json",
       track="rstats", tweets=10, oauth=my_oauth )

  ## capture tweets published by Twitters official account
    filterStream( file.name="tweets_twitter.json",
       follow="783214", timeout=600, oauth=my_oauth )

  ## capture tweets sent from New York City in Spanish only, and saving as an object in memory
    tweets <- filterStream( file.name="", language="es",
        locations=c(-74,40,-73,41), timeout=600, oauth=my_oauth )

  ## capture tweets mentioning the "rstats" hashtag or sent from New York City
    filterStream( file="tweets_rstats.json", track="rstats",
        locations=c(-74,40,-73,41), timeout=600, oauth=my_oauth )

  ## capture 100 tweets sent from New York City and storing in MongoDB, in collection
  ## nyc of database tweets
    tweets <- filterStream( ns="tweets.nyc",
        locations=c(-74,40,-73,41), tweets=100, oauth=my_oauth )

  ## same as above, but also storing tweets in disk
    tweets <- filterStream( file.name="tweets_nyc.json", ns="tweets.nyc",
        locations=c(-74,40,-73,41), tweets=100, oauth=my_oauth )

  ## End(Not run)
```

`getTweets` *Connect to Mongo database and extract tweets that match conditions specified in the arguments.*

Description

`getTweets` opens a connection to a Mongo database and returns tweets that match a series of conditions: whether it contains a certain keyword, whether it is or not a retweet, or whether or not it contains a hashtag. It allows to specify the fields of the tweet to be extracted. The function depends on the rmongodb package, and a mongo object needs to be loaded in user's workspace.

Usage

```
getTweets(ns, string = NULL, df = TRUE,
  fields = c("created_at", "user.screen_name", "text"),
  retweets = NULL, hashtags = NULL, verbose = TRUE)
```

Arguments

`ns`	string, namespace of the MongoDB collection where tweets were stored. Generally, it will be of the form "database.collection".
`string`	string, set to NULL by default (will return count of all tweets). If it is a string, it will return the number of tweets that contain that string in the text of the tweet.
`df`	logical, default is `TRUE`, which will return tweets in a data frame. Otherwise, tweets will be return in list format.
`fields`	vector of strings, indicates fields from tweets that will be returned. Default is the date and time of the tweet, its text, and the screen name of the user that published it. See details for full list of possible fields.
`retweets`	logical, set to NULL by default (will return count of all tweets). If `TRUE`, will count only tweets that are retweets (i.e. contain an embededed retweeted status - manual retweets are not included). If `FALSE`, will count only tweets that are not retweets (manual retweets are now included).
`hashtags`	logical, set to NULL by default (will return count of all tweets). If `TRUE`, will count only tweets that use a hashtag. If `FALSE`, will count only tweets that do not contain a hashtag.
`verbose`	logical, default is `TRUE`, which generates some output to the R console with information about the count of tweets.

Details

The following is a non-exhaustive of relevant fields that can be specified on the `fields` argument (for a complete list, check the documentation at: `https://dev.twitter.com/docs/platform-objects` Tweet: text, created_at, id_str, favorite_count, source, retweeted, retweet_count, lang, in_reply_to_status_id, in_reply_to_screen_name Entities: entities.hashtags, entities.user_mentions, entities.hashtags, entities.urls Retweeted_status: retweeted_status.text, retweeted_status.created_at... (and all other tweet, user, and entities fields) User: user.screen_name, user.id_str, user.geo_enabled, user.location, user.followers_count, user.statuses_count, user.friends_count, user.description, user.lang, user.name, user.url, user.created_at, user.time_zone Geo: geo.coordinates

Author(s)

Pablo Barbera <`pablo.barbera@nyu.edu`>

Examples

```
## Not run:
## capture 100 tweets sent from New York City and stored in local MongoDB,
## in collection nyc of database tweets
 load(my_oauth)
 tweets <- filterStream( ns="tweets.nyc",
      locations=c(-74,40,-73,41), tweets=100, oauth=my_oauth )

## connect to the Mongo database using rmongodb package
 library(rmongodb)
 mongo <- mongo.create("localhost", db="tweets")
## if required, specify username and password
## (MongoDB defaults are empty username and password)
 mongo.authenticate(mongo, username="", password="",
   db="tweets")
```

```
## extract text from all tweets in the database
 tweets <- getTweets( ns="tweets.nyc", fields="text")

## extract tweets, with text and screen name
 tweets <- getTweets( ns="tweets.nyc", fields=c("user.screen_name", "text"))

## extract all tweets that mention times square
 tweets <- getTweets( ns="tweets.nyc", string="times square")

## extract date of all tweets that mention times square
 tweets <- getTweets( ns="tweets.nyc", string="times square", fields="created_at")

## End(Not run)
```

parseTweets *Converts tweets in JSON format to data frame.*

Description

This function parses tweets downloaded using `filterStream`, `sampleStream` or `userStream` and returns a data frame.

Usage

```
parseTweets(tweets, simplify = FALSE, verbose = TRUE)
```

Arguments

`tweets`	A character string naming the file where tweets are stored or the name of the object in memory where the tweets were saved as strings.
`simplify`	If `TRUE` it will return a data frame with only tweet and user fields (i.e., no geographic information or url entities).
`verbose`	logical, default is `TRUE`, which will print in the console the number of tweets that have been parsed.

Details

`parseTweets` parses tweets downloaded using the `filterStream`, `sampleStream` or `userStream` functions and returns a data frame where each row corresponds to one tweet and each column represents a different field for each tweet (id, text, created_at, etc.).

The total number of tweets that are parsed might be lower than the number of lines in the file or object that contains the tweets because blank lines, deletion notices, and incomplete tweets are ignored.

To parse json to a twitter list, see `readTweets`. That function can be significantly faster for large files, when only a few fields are required.

Author(s)

Pablo Barbera <`pablo.barbera@nyu.edu`>

See Also

filterStream, sampleStream, userStream

Examples

```
## The dataset example_tweets contains 10 public statuses published
## by @twitterapi in plain text format. The code below converts the object
## into a data frame that can be manipulated by other functions.

data(example_tweets)
tweets.df <- parseTweets(example_tweets, simplify=TRUE)

## Not run:
## A more complete example, that shows how to capture a users home timeline
## for one hour using authentication via OAuth, and then parsing the tweets
## into a data frame.

 library(ROAuth)
 reqURL <- "https://api.twitter.com/oauth/request_token"
 accessURL <- "http://api.twitter.com/oauth/access_token"
 authURL <- "http://api.twitter.com/oauth/authorize"
 consumerKey <- "xxxxxyyyyyzzzzzz"
 consumerSecret <- "xxxxxxyyyyyzzzzzzz111111222222"
 my_oauth <- OAuthFactory$new(consumerKey=consumerKey,
                              consumerSecret=consumerSecret,
                              requestURL=reqURL,
                              accessURL=accessURL,
                              authURL=authURL)
 my_oauth$handshake()
 userStream( file="my_timeline.json", with="followings",
         timeout=3600, oauth=my_oauth )
 tweets.df <- parseTweets("my_timeline.json")

## End(Not run)
```

---

readTweets *Converts tweets in JSON format to R list.*

Description

This function parses tweets downloaded using `filterStream`, `sampleStream` or `userStream` and returns a list.

Usage

```
readTweets(tweets, verbose = TRUE)
```

Arguments

`tweets`	A character string naming the file where tweets are stored or the name of the object in memory where the tweets were saved as strings.
`verbose`	logical, default is `TRUE`, which will print in the console the number of tweets that have been parsed.

Details

This function is the first step in the parseTweets function and is provided now as an independent function for convenience purposes. In cases where only one field is needed, it can be faster to extract it directly from the JSON data read in R as a list. It can also be useful to extract fields that are not parsed by parseTweets, such as hashtags or mentions.

The total number of tweets that are parsed might be lower than the number of lines in the file or object that contains the tweets because blank lines, deletion notices, and incomplete tweets are ignored.

Author(s)

Pablo Barbera <pablo.barbera@nyu.edu>

See Also

parseTweets.

Examples

```
## The dataset example_tweets contains 10 public statuses published
## by @twitterapi in plain text format. The code below converts the object
## into a list and extracts only the text.

data(example_tweets)
tweets.list <- readTweets(example_tweets)
only.text <- unlist(lapply(tweets.list, [[, text))
## it can be done with an explicit loop:
only.text <- c()
for (i in 1:length(tweets.list)){
   only.text[i] <- tweets.list[[i]][text]
}
print(unlist(only.text))
```

---

sampleStream   *Connect to Twitter Streaming API and return a small random sample of all public statuses.*

Description

sampleStream opens a connection to Twitter's Streaming API that will return a small random sample of public statuses, around 1% at any given time.

Usage

```
sampleStream(file.name, timeout = 0, tweets = NULL,
  oauth = NULL, verbose = TRUE)
```

Arguments

file.name	string, name of the file where tweets will be written. "" indicates output to the console, which can be redirected to an R object. If the file already exists, tweets will be appended (not overwritten).
timeout	numeric, maximum length of time (in seconds) of connection to stream. The connection will be automatically closed after this period. For example, setting `timeout` to 10800 will keep the connection open for 3 hours. The default is 0, which will keep the connection open permanently.
tweets	numeric, maximum number of tweets to be collected when function is called. After that number of tweets have been captured, function will stop. If set to `NULL` (default), the connection will be open for the number of seconds specified in `timeout` parameter.
oauth	an object of class `oauth` that contains the access tokens to the user's twitter session. This is currently the only method for authentication. See examples for more details.
verbose	logical, default is `TRUE`, which generates some output to the R console with information about the capturing process.

Details

For more information, check the documentation at: `https://dev.twitter.com/docs/api/1.1/get/statuses/sample`

Note that when no file name is provided, tweets are written to a temporary file, which is loaded in memory as a string vector when the connection to the stream is closed.

The total number of actual tweets that are captured might be lower than the number of tweets requested because blank lines, deletion notices, and incomplete tweets are included in the count of tweets downloaded.

Author(s)

Pablo Barbera <`pablo.barbera@nyu.edu`>

See Also

`filterStream`, `userStream`, `parseTweets`

Examples

```
## Not run:
## capture a random sample of tweets
sampleStream( file.name="tweets_sample.json", user=FOO, password=BAR )

## An example of an authenticated request using the ROAuth package,
## where consumerkey and consumer secret are fictitious.
## You can obtain your own at dev.twitter.com
 library(ROAuth)
 reqURL <- "https://api.twitter.com/oauth/request_token"
 accessURL <- "http://api.twitter.com/oauth/access_token"
 authURL <- "http://api.twitter.com/oauth/authorize"
 consumerKey <- "xxxxxyyyyyzzzzzz"
 consumerSecret <- "xxxxxxyyyyyzzzzzzz111111222222"
  my_oauth <- OAuthFactory$new(consumerKey=consumerKey,
    consumerSecret=consumerSecret, requestURL=requestURL,
    accessURL=accessURL, authURL=authURL)
 my_oauth$handshake(cainfo = system.file("CurlSSL", "cacert.pem", package = "RCurl"))
 sampleStream( file.name="tweets_sample.json", oauth=my_oauth )


## End(Not run)
```

`topRetweets` *Connect to Mongo database and extract top retweets that match conditions specified in the arguments.*

Description

`topRetweets` opens a connection to a Mongo database and returns all retweets (or only retweets that mention a specific keyword) ordered by total number of retweets received.

Usage

```
topRetweets(ns, string = NULL, min = 10, verbose = TRUE)
```

Arguments

`ns`	string, namespace of the MongoDB collection where tweets were stored. Generally, it will be of the form "database.collection".
`string`	string, set to NULL by default (will return count of all tweets). If it is a string, it will return the number of tweets that contain that string in the text of the tweet.
`min`	numeric, set to 10 by default (will return all retweets whose retweet count is at least 10). In large datasets, choose a high number to increase speed of query.
`verbose`	logical, default is `TRUE`, which generates some output to the R console with information about the count of tweets.

Details

Note that this function will only return retweets that are made using the built-in retweeting system - this is, 'manual' retweets using copy&paste are not included. Also note that total retweet counts are based on Twitter's internal tally, and do not reflect the number of retweets in the database. In other words, it could happen that the most popular retweet in a given moment is a tweet that was originally sent days ago, but was retweeted during the time of that tweets were captured.

Author(s)

Pablo Barbera <`pablo.barbera@nyu.edu`>

Examples

```
## Not run:
## capture 100 tweets that mention "twitter" and store them in local MongoDB,
## in collection twitter of database tweets
## (Note: since the track parameter searches also in URLs, this search
## is equivalent to capturing only tweets that contain pictures sent
## using Twitters built-in service)
```

```
 load(my_oauth)
 tweets <- filterStream( ns="tweets.twitter",
      track="twitter", tweets=100, oauth=my_oauth )

## connect to the Mongo database using rmongodb package
 library(rmongodb)
 mongo <- mongo.create("localhost", db="tweets")
## if required, specify username and password
## (MongoDB defaults are empty username and password)
 mongo.authenticate(mongo, username="", password="",
   db="tweets")

## extract all retweets that were retweeted at least 100 times
 rts <- topRetweets( ns="tweets.twitter", min=100)

## show top 10 retweets from previous query
 head(rts, n=10)

## End(Not run)
```

`userStream` *Connect to Twitter Streaming API and return messages for a single user.*

Description

`userStream` opens a connection to Twitter's Streaming API that will return statuses specific to the authenticated user. The output can be saved as an object in memory, written to a text file or stored in MongoDB.

Usage

```
userStream(file.name = NULL, with = "followings",
  replies = NULL, track = NULL, locations = NULL,
  timeout = 0, tweets = NULL, oauth, ns = NULL,
  host = "localhost", username = "", password = "",
  verbose = TRUE)
```

Arguments

`file.name`	string, name of the file where tweets will be written. "" indicates output to the console, which can be redirected to an R object. If the file already exists, tweets will be appended (not overwritten).
`with`	string, detault is "followings", which will stream messages from accounts the authenticated user follow. If set to "user", will only stream messages from authenticated user. See the `with` parameter information in the Streaming API documentation for details: https://dev.twitter.com/docs/streaming-apis/parameters#with
`replies`	string, default is `NULL`, which will only stream replies sent by a different user if the authenticated user follows the receiver of the reply. All replies to users that the authenticated user follows will be included if this argument is set to "all". See the `replies` parameter information in the Streaming API documentation for details: https://dev.twitter.com/docs/streaming-apis/parameters#replies

`track`	string or string vector containing keywords to track. See the track parameter information in the Streaming API documentation for details: `http://dev.twitter.com/docs/streaming-apis/parameters#track`.
`locations`	numeric, a vector of longitude, latitude pairs (with the southwest corner coming first) specifying sets of bounding boxes to filter statuses by. See the locations parameter information in the Streaming API documentation for details: `http://dev.twitter.com/docs/streaming-apis/parameters#locations`
`timeout`	numeric, maximum length of time (in seconds) of connection to stream. The connection will be automatically closed after this period. For example, setting `timeout` to 10800 will keep the connection open for 3 hours. The default is 0, which will keep the connection open permanently.
`tweets`	numeric, maximum number of tweets to be collected when function is called. After that number of tweets have been captured, function will stop. If set to `NULL` (default), the connection will be open for the number of seconds specified in `timeout` parameter.
`oauth`	an object of class `oauth` that contains the access tokens to the user's twitter session. This is the only method for authentication available for user streams. See examples for more details.
`ns`	string, namespace of the collection to which tweets will be added. Generally, it will be of the form "database.collection". If the database or the collection do not exist, they will be automatically created; if they exist, tweets will be appended.
`host`	string host/port where mongo database is hosted. Default is localhost (127.0.0.1).
`username`	string, username to be used for authentication purposes with MongoDB.
`password`	string, password corresponding to the given username.
`verbose`	logical, default is `TRUE`, which generates some output to the R console with information about the capturing process.

Details

This function provides access to messages for a single user.

The set of messages to be returned can include the user's tweets and/or replies, and public statuses published by the accounts the user follows, as well to replies to those accounts.

Tweets can also be filtered by keywords and location, using the `track` and `locations` arguments.

The total number of actual tweets that are captured might be lower than the number of tweets requested because blank lines, deletion notices, and incomplete tweets are included in the count of tweets downloaded.

Note that when no file name is provided, tweets are written to a temporary file, which is loaded in memory as a string vector when the connection to the stream is closed.

To store tweets in MongoDB, it is necessary to install the MongoDB server in a local or remote machine. See here for instructions: `http://docs.mongodb.org/manual/installation/`

Author(s)

Pablo Barbera <`pablo.barbera@nyu.edu`>

See Also

`filterStream`, `sampleStream`, `parseTweets`

Examples

```
## Not run:
## The following example shows how to capture a users home timeline
## with the Streaming API and using authentication via the ROAuth
## package, with fictitious consumerkey and consumer secret.
## You can obtain your own at dev.twitter.com
 library(ROAuth)
 requestURL <- "https://api.twitter.com/oauth/request_token"
 accessURL <- "http://api.twitter.com/oauth/access_token"
 authURL <- "http://api.twitter.com/oauth/authorize"
 consumerKey <- "xxxxxyyyyyzzzzzz"
 consumerSecret <- "xxxxxxyyyyyzzzzzzz111111222222"
 my_oauth <- OAuthFactory$new(consumerKey=consumerKey,
    consumerSecret=consumerSecret, requestURL=requestURL,
    accessURL=accessURL, authURL=authURL)
 my_oauth$handshake(cainfo = system.file("CurlSSL", "cacert.pem", package = "RCurl"))
 save(my_oauth, file="my_oauth")
 userStream( file.name="my_timeline.json", with="followings",
     timeout=600, oauth=my_oauth )
## Capturing 10 tweets from users timeline and storing in MongoDB
load("my_oauth")
 userStream( ns="tweets.mytimeline", with="followings",
     tweets=10, oauth=my_oauth )

## End(Not run)
```

Index

∗Topic **datasets**
    example_tweets, 3

countTweets, *2*, 2

example_tweets, 3

filterStream, *2*, 4, *8, 9*, *11*, *14*

getTweets, *2*, 6

parseTweets, *2*, *5*, 8, *10*, *11*, *14*

readTweets, *2*, *8*, 9

sampleStream, *2*, *5*, *8*, *9*, 10, *14*
streamR *(streamR-package)*, 2
streamR-package, 2

topRetweets, *2*, 12

userStream, *2*, *5*, *8*, *9*, *11*, 13